UPPSALA
UNIVERSITET

Självständigt arbete i informationsteknologi
19 juni 2017

User Configurable Indoor Positioning System using WiFi Trilateration and Fingerprinting

Anton Carlsson
Filip Gölander
Fredrik Sandelin

Civilingenjörsprogrammet i informationsteknologi

Computer and Information Engineering Programme

UPPSALA
UNIVERSITET

Institutionen för informationsteknologi

Besöksadress:
ITC, Polacksbacken
Lägerhyddsvägen 2

Postadress:
Box 337
751 05 Uppsala

Hemsida:
http://www.it.uu.se

Abstract

User Configurable Indoor Positioning System using WiFi Trilateration and Fingerprinting

Anton Carlsson
Filip Gölander
Fredrik Sandelin

The use of smartphones for positioning and navigation is mostly limited to outdoor settings. Indoors, where GPS signals are too inaccurate for positioning, an alternative must be used. This project aimed at producing an indoor positioning system which could be both configured and used by an end user organization without equipping its buildings with proprietary hardware. The prerequisites were that a complete digital representation of the building floors is available, and that the floors have a sufficient amount of WiFi access points. Our system measures radio signal strength from existing WiFi infrastructure using a smartphone. This data is sent to a backend and is used to position a device using two different methods: trilateration and fingerprinting. The finished system can position a user with an accuracy of approximately four meters using fingerprinting instead of trilateration as it yielded the best results. The building used for testing was scanned using a smartphone equipped with our application, something that we would expect an end user to be able to do.

Handledare: Virginia Grande Castro och Björn Victor
Examinator: Björn Victor

Sammanfattning

Användandet av smartphones för positionering och navigering är mestadels inriktat på utomhusanvändning. Inomhus är GPS-signaler inte tillräckligt noggranna för positionering, och ett alternativ måste användas. Det här projektets mål var att producera ett inomhuspositioneringssystem som kan konfigureras och användas av slutanvändarorganisationen utan att behöva utrusta sina byggnader med proprietär hårdvara. Förutsättningarna är att en komplett digital representation av byggnaden finns tillgänglig, och att våningsplanen har tillräckligt många WiFi-basstationer. Vårt system mäter radiosignalstyrka från den existerande WiFi-infrastrukturen. Denna data skickas till en backend och används i två olika metoder: trilateration och fingerprinting. Det slutgiltiga systemet kan positionera en användare med en träffsäkerhet på ungefär fyra meter när fingerprintingmetoden används då den producerade det bästa resultatet. Byggnaden som systemet testades i skannades av en smartphone med vår applikation, en sak som vi förväntar oss att en slutanvändare skulle kunna göra själv.

Contents

List of Figures

List of Tables

List of Equations

Words and abbreviations.

RSSI Received signal strength indicator.

WiFi The 802.11 wireless local area network standard.

AP WiFi access point.

IDE Integrated development environment.

BSSID Basic service set identifier, the unique identifier of a network.

API Application programming interface.

GPS Global positioning system.

BLE Bluetooth low energy.

Position (verb) The process of finding x, y coordinates of a given device, person or object.

NLOS Non-line-of-sight.

PLE Path Loss Exponent.

Radio map A dataset containing BSSID and RSSI measurement data as well as the respective position in a building at which the measurements were made.

JS JavaScript

XML Extensible Markup Language

dBm decibel-milliwats

1 Introduction

Since the release of the Apple iPhone 3G in 2008 [1], navigation has been available in iPhones. This caused navigation to be widely available in all smartphones thanks to iPhone's significant market share [2]. However, the navigation feature has mostly been limited to outdoor use since GPS and its siblings do not accurately work when the path between the device and the satellite is partially or completely blocked [3].

As for available indoor positioning system, a large number of them are dependent on proprietary hardware such as BLE beacons (see Section 4). This method involves installing extra hardware, which is both expensive and time-consuming. Another problem, which applies to all indoor positioning systems, is that the systems need some sort of digital representation of the building, a prerequisite that is also expensive and time-consuming to fulfill. Because of this, we have built a WiFi based indoor positioning system that integrates with an existing system containing digital representations of buildings. Our system can be configured and used without installing proprietary hardware in the building in question, given that the building has a WiFi infrastructure.

The finished system consists of an Android application in which users can calibrate and configure buildings that are deployed in the system. The Android application also displays where in the building the device on which it is running is located by sending WiFi signal data to web services. These web services compare the gathered data with already known data of the building and calculate the position of the device in question by utilizing two different positioning algorithms. The calculated position is displayed in the mobile application on a map of the building.

The first method, fingerprinting[1], was able to position a Samsung Galaxy S6 smartphone with an accuracy of four meters 85% to 90% of the time, depending on the age of the building's fingerprint. The other method, trilateration[2], proved to be less accurate, being able to position the same device with a four meter accuracy 50% of the time. Although more inaccurate, it should be noted that the facility in which experiments were conducted lacked complete signal coverage from known WiFi access points (APs), which is essential for trilateration.

[1] Explained in Section 2.5
[2] Explained in Section 2.2

2 Background

There are several techniques which are used for indoor positioning. They can be used alone or in conjunction with each other. This section intends to give an overview of existing methods for positioning and the corporate partner for the project: Pythagoras AB.

2.1 Pythagoras AB

The external stakeholder for this project is Pythagoras AB [4] (hereby Pythagoras), a software company based in Stockholm. They have developed a system in which the customers' buildings are digitally represented. This includes floor plans, 3D models of the building, information about departments and any contracts active in them. Almost any component of a building can be modeled: for example, plants, furniture and network access points. Attributes can be added to components containing any information necessary for a useful digital representation of that component such as expiration date of a fire extinguisher or acquisition date of a coffee machine. The tool is called SpaceManager and is used for creating and modifying the digital representation as described above. SpaceManager can be used by both Pythagoras and their customers.

Pythagoras' system as a whole is intended for administrative purposes and is marketed as a tool for increasing facility efficiency. Accompanying this system is an API (Application Programming Interface) used internally in order to get data about a company's buildings. Pythagoras' interest in the project was based on the wish to offer positioning capabilities for their customers buildings without the use of additional hardware.

2.2 Trilateration

Trilateration is a method used to determine the relative position of a point in space. The method is most easily described using images. In Figure 1 the method is visualized in two-dimensional space. It can be simplified as follows: draw a circle around each known point (p_1, p_2, p_3), with the radius as the known distance to them from p (r_1, r_2, r_3). The intersection of these circles will be the point, p, we wish to locate. The method works in the same manner when applied in three dimensions, but doing so results in replacing the circles with spheres and requiring a fourth known distance in order to determine the position of a point.

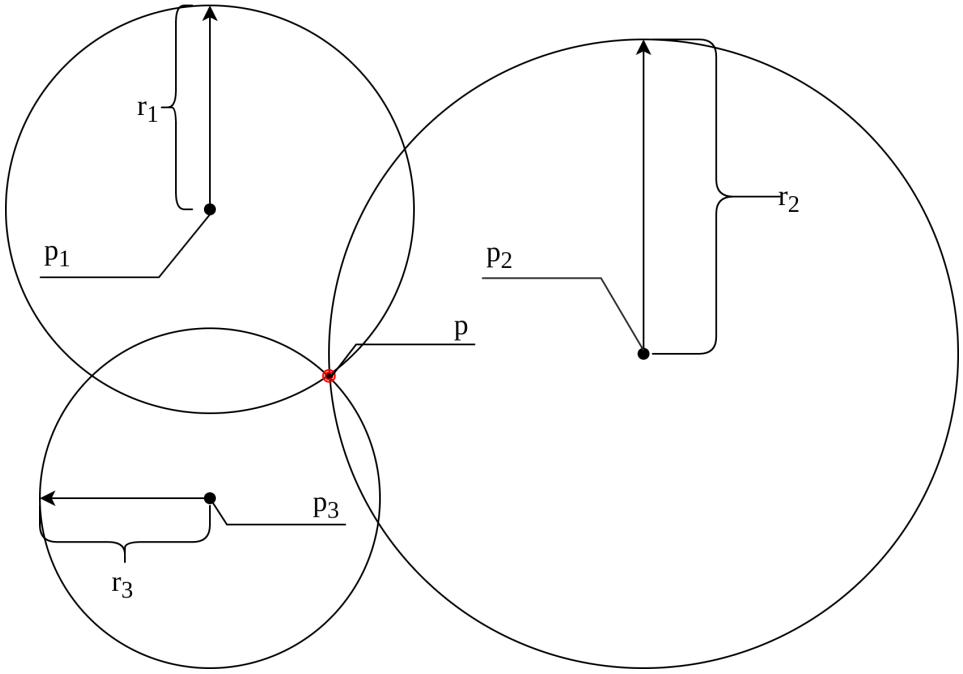

Figure 1: Visualization of trilateration algorithm.

2.3 Received signal strength indication

Received signal strength indication (RSSI) is a measurement of received signal strength. The standard for WiFi: 802.11, standardized by the Institute of Electrical and Electronics Engineers (IEEE), includes the measurement as an optional parameter [5]. The value is intended to be relative and an absolute accuracy is not specified, meaning the vendor of the hardware can choose what the value represents. In the hardware specification, it is to be increasing from zero to a predefined upper limit. In the Android operating system, used in this project, it is given in the unit dBm [6].

2.4 Signal propagation

Signal propagation is the behavior of signals as they propagate (travel) through the air, and can be described by a signal propagation model. Power based WiFi positioning approaches can use a signal propagation model called a path loss model to describe the relationship between the power attenuation of the signal and the distance between the transmitter and the receiver [7]. Power attenuation is the gradual loss of power as

the signal propagates. Objects around the building can cause further power attenuation because of shadow fading, which is the variation in power attenuation due to objects affecting the signal propagation.

2.5 Fingerprinting

Fingerprinting is a machine learning method which uses prerecorded radio data to determine a user's position. It consists of two phases: the offline phase and the online phase. In the offline phase, known positions are visited and scanned, thereby acquiring the different APs' RSSI values with a specific location as reference and storing them in a database. The database forms a fingerprint of the building, often referred to as a radio map. In the online phase, also known as the tracking phase, a device is positioned. The device measures RSSI for all the APs in range, queries the database and determines the best matching position. The techniques used for determining the best matching position can vary from simple machine learning algorithms to more complex probabilistic models [8, 9, 10, 11, 12]. These techniques and methods are discussed in Section 5.6.

2.6 Global Positioning System

Global Positioning System (GPS) is a positioning system which is owned and operated by the United States government [13]. It consists of a network of satellites orbiting the Earth. At regular intervals, each satellite broadcasts its own position and the current time. A device which is to be positioned receives this signal and calculates the distance to the satellite using the time difference between the time that the message was received and the time of broadcast. Knowledge of four such distances allows the device to use trilateration to calculate its position in three-dimensional space. The accuracy of the system in a smartphone is typically around five meters under open sky. However, statistics indicate that the accuracy is often far better [3]. The accuracy quickly decreases when the path between device and satellite is obstructed, which is why GPS is less suitable for indoor environments [3].

2.7 Bluetooth Low Energy

Bluetooth Low Energy (BLE) is a part of the Bluetooth specification. It was added to the specification with Bluetooth version 4.0. Devices compliant with this part of the specification consume very little energy. The consumption is so low that they can operate for months or years on only consumer grade disposable batteries [14]. In this

paper we refer to small, battery powered devices utilizing this technology to provide various services as BLE beacons.

3 Purpose, aims, and motivation

This project strives to provide a cost-effective, easy-to-deploy, indoor positioning system which is integrated into the facility management system in use at Pythagoras. The inexpensive nature of the system makes it attractive for user organizations to adopt. A system which is easy to deploy increases the variety of ventures interested in it. By providing the core functionality for positioning many other applications can be built on top of the system, such as navigation, personnel traffic analyzers and cleaning schedules for facilities. Such applications would be able to relieve time and energy for users of the system.

The aim of the project is to deliver a system that can position a smartphone in a building with moderate accuracy. Pythagoras wishes to have an accuracy corresponding to that of a normal sized room. This, in its essence, means that if the smartphone is in a certain room, the system will not perceive it to be in any other room but that one. The project group's personal aim is to have an accuracy of four meters, because we judge that to be the size of a normal sized room. In the process of achieving our own aim, we hope to satisfy Pythagoras' specification. Other more general goals of this project is to answer the following questions: How can the system be easily configured by a customer? How accurate can a user configured WiFi-only positioning system be?

The motivation to provide inexpensive indoor positioning comes from the desire to offer smart building features to companies and organizations. Target organizations are those which do not want to spend money on buying, installing and maintaining proprietary hardware. In Section 4 other systems on the market are described.

There already exists positioning systems based on WiFi signals. However, configuring these with information of where APs are located is tedious and time-consuming. By integrating with Pythagoras' system we can achieve a system that runs on existing infrastructure and where the customers can easily insert the location of access points in their facilities. In addition to the customer configuration, we will use Pythagoras' system to build a fingerprinting module where customers themselves can measure RSSI values in their building to produce even more accurate positioning.

3.1 Delimitations

Because of time constraints, this project was meant to be a proof of concept for indoor positioning using WiFi rather than a final implementation used in Pythagoras' system. Therefore, little to no security concerns have been taken into consideration and the scope of this project is limited to only WiFi positioning. GPS was not used in combination with RSSI to get better accuracy but was used in order to get the correct building schematics to use. This decision was made since combining the positioning methods was considered too time-consuming for the size of this project. Furthermore, the goal of user configurability only extends to what could be technically achieved with a smartphone and not how the user interface could generate a better experience for configuration.

To avoid maintaining several code bases, we chose to only implement the system on one operating system, Android. We chose Android because it has the highest market share [15] and allow WiFi scanning. Our application uses the Android API level 15, which means that the oldest Android version needed to run our application is Ice Cream Sandwich (4.0.3+). Since this includes 98.2% of all Android smartphones, including a lower version of the API was not prioritized [16].

4 Related work

In this section we examine and explain what some relevant works are and how they work. According to our investigations, an overwhelming majority of commercial solutions for indoor positioning, partially or completely, use BLE beacons [17, 18, 19, 20, 21, 22]. This might be a result of Apple's decision to not allow WiFi scanning on applications on the AppStore since WiFi scanning is a part of their private API [23].

4.1 BLE beacons

Estimote [17] is a commercial solution available for purchase. It has no support for WiFi and instead uses BLE technology, which means that beacons have to be purchased, deployed and eventually have their batteries replaced. The beacons' battery life are in the span of one to five years according to Estimote's website. Estimote does not to seem to provide a digital representation of the customer's building. Essentially the system can only measure where users are in relation to the beacons. How users change where beacons are perceived to be located by the system is not disclosed. Estimote aims for

the same end result as this project does: indoor positioning. However, it is dependent on deploying proprietary hardware, which is something we attempt to avoid.

Estimote does not disclose which specific positioning algorithms it uses in conjunction with the beacons. Several of the systems below also use BLE. However, more information is available on how the beacons are used, which is why they are discussed in the more specific sections below.

4.2 Trilateration

Pathak et al. researched the viability of trilateration algorithms for indoor navigation [24]. For this they created an Android application that measured the RSSI of nearby WiFi networks. This data was sent to a server which calculated the position. Their average margin of error was 1.99 meters. However, their test case was constrained. They only tested a small number of positions (six) on a single floor. The system proposed only consisted of an application that sent data to a server which returned a position to a custom built map. It did not integrate with a map service as our project aims to do. User configuration is not discussed in the paper.

CampusGuiden was a company providing an indoor positioning system using WiFi trilateration [25]. However, when they rebranded to MazeMap they stopped providing their own positioning system. The system they sell now only integrates with existing ones. Their service was used at Trondheim University's campus. Unfortunately the accuracy of the system is not disclosed.

4.3 Fingerprinting

Accuware is one of the indoor positioning systems currently available as a commercial product [26]. It uses a combination of WiFi fingerprinting and BLE beacons to determine the position of the user. Combining these two techniques they manage to get an accuracy of one meter or less [27].

Insiteo is a company that provides an indoor positioning system [28]. This system can estimate a user's position within two to three meters of their real position [29]. It utilizes WiFi and BLE beacon fingerprinting, accelerometer and compass to achieve that accuracy.

Haverinen et al. explored the capabilities of using the magnetic field fluctuations present inside buildings as a tool for indoor positioning [30]. The Earth's magnetic field is

affected by man-made structures such as steel door frames and reinforced concrete. The result of this is, according to Haverinen et al. , that each building has a unique magnetic fingerprint. This fingerprint can be mapped in one or two dimensions using a three-dimensional magnetometer. The resulting map is then superimposed on a floor map, on which a user can be positioned by again using the magnetometer and comparing the values to the map. Not all three axis of the magnetometer is needed for positioning. However, using all of them will yield the best result.

IndoorAtlas provides a cloud-based platform for indoor positioning [31]. They utilize the magnetometer present in today's smartphones to detect anomalies in Earth's geomagnetic field. The company was founded by Janne Haverinen, one of the authors of the paper mentioned in the previous paragraph, and claims to have a one to two meter accuracy.

Although magnetic positioning is accurate and can be configured by a smartphone, the requirement is that the buildings have a distinguishable magnetic fingerprint. In buildings mostly comprised of wood, the precision decreases [32] since the magnetic variations that make positioning possible stems from steel frames and reinforced concrete [30].

4.4 Dead reckoning

Beauregard et al. investigated positioning a pedestrian by means of dead reckoning [33]. This technique estimates a user's position based on the distance traveled in a specific direction from a known position. An acceleration sensor was used to determine when footsteps were taken, and a GPS receiver in conjunction with a compass were used to determine direction. A distance was estimated by combining the number of footsteps taken with the approximated length of a footstep. Their results showed an accumulated error of 2% over a four kilometer walk. The acceleration sensor was mounted on a helmet and the author acknowledged that if the sensor was to be placed in a more loose manner, such as in a pocket or backpack, the results are expected to be "distinctly worse" [33]. As mentioned, the system used GPS to determine direction. However, if other methods were to be used for this, the technique could be used for coarse indoor positioning.

5 Method

This section aims to explain the methods and techniques used for developing the system. This includes both development choices as well as algorithms related to positioning. We will present alternatives and explain our reasoning behind the choices we made.

5.1 Android

The user interacts with an Android application that communicates with a Java backend server. The alternative to Android would have been iOS or Windows Mobile. Apple does not currently provide a way to access the RSSI of all nearby WiFi networks through the public API, thereby removing the possibility of deploying an application in the AppStore. The market share for Windows Phone is, at the moment, extremely slim [15] which is why that was not an option either. The application communicates with the backend using a RESTful API, an API style that allows fetching and manipulation of web resources through stateless operations, meaning user sessions do not need to be tracked.

5.2 Web frontend

The web frontend used for displaying the users current position on a map was intentionally made simple. The reason for this is that the system was intended to be further integrated into Pythagoras' existing system. This includes mobile applications and other frontend software which have better interfaces than we could produce in the given time frame. The standards HTML and CSS were used in conjunction with the scripting language JavaScript (JS) to develop the web part of the frontend, as well as the tools described below.

JQuery JQuery is a JavaScript library made to ease the development of JavaScript applications by making tasks often performed using JavaScript easier [34]. Main features include enhanced HTML traversal, event handling, Asynchronous JavaScript and XML (AJAX), and animation [34]. The main use in this project has been AJAX, which is the way the web application communicates with the backend server, and HTML traversal and manipulation. There are smaller libraries which include only the features which are necessary for this project. However, since Pythagoras uses JQuery in their existing applications, it was chosen to ease further integration with their system in the future.

Scalable Vector Graphics SVG is a markup language used for defining two-dimensional vector graphics. It is based on the XML standard and it is supported by most modern browsers [35]. SVG was developed by the W3C SVG Working Group and is still being maintained by it. The latest complete version, and the one which is recommended by the maintaining group, is version 1.1. SVG is the format in which floor plans are delivered by the API described in Section 2.1, which is the reason it was used. In order to manipulate the SVG files, e.g. to add markers to indicate a users position, the JavaScript library Snap.svg was used [36].

5.3 RESTful Web Services

RESTful web services are web services which adhere to the REST design principles. Core concepts of the REST design that are important in our system are: simple clients, uniform accessibility and cacheable calls [37]. This means that we can keep clients simple by extending complex functionality from the RESTful API instead of increasing the complexity of the client. Uniform accessibility means that all clients can use the same API. Cacheable calls are calls that let the client store server responses. The result of this is that if the client tries to make another identical call it can utilize the saved response instead of making a redundant call. RESTful was chosen as we find these three concepts important for future expansions and scaling of the system.

The web services developed in this project were built using Jersey [38], an open source Java framework designed to produce such services. Several programming languages capable of sending and receiving HTTP requests were considered: for example, Python, Ruby and Java. However, since this system is intended to be integrated with Pythagoras' facility management softwares, which are written in Java, the choice was made to develop it in Java.

An alternative to RESTful web services is SOAP web services. SOAP is, in contrast to REST, a complete protocol and is completely based on Extensible Markup Language (XML) [39], whereas REST supports both JavaScript object notation (JSON) and XML. SOAP would therefore require the client to implement XML parsing [40]. Since the communication will be between JS and Java, RESTful is a better alternative. This is because JSON is based on the object format that JavaScript uses. JSON parsing is also natively supported by the JavaScript engine [40], responsible for interpreting and running the code. As Pythagoras services are deployed through RESTful web services and since JSON eases the use of the data being transmitted between backend and frontend, the decision was made to use RESTful services using JSON as the data transfer format.

5.4 Signal propagation

Signal propagation is explained in Section 2.4. Oguejiofor et al. proposed a general log-distance path loss model for non-line-of-sight (NLOS) propagation that uses the difference in dBm values to calculate the distance between the transmitter and the receiver [41]. The difference is calculated with the RSSI value measured at the unknown position and the reference value, measured one meter from the AP. Akl et al. added a Gaussian distributed random variable to their model to account for log-normal shadowing [42] (explained in Section 2.4). We combined these two models for the basis of our own, seen in Equation 1, Section 10.1.

However, our model could be improved. Bose et al. [7] mentioned the potential use of the Hata-Okumara model, a more intricate model for signal propagation which takes into account a larger number of variables, for example antenna gain. For the most optimal results one should empirically calculate the path loss exponent (PLE) [41, 42, 7, 43] and the standard deviation of the random variable [42, 43]. For further optimization it is possible to use unique variables for each AP [42], or use multiple PLEs together with a signal strength threshold [7]. This was not implemented because of time constraints.

5.5 Trilateration

WiFi trilateration uses distances to calculate an intersect point or area, as explained in Section 2.2. This required measured RSSI values to be translated into a unit of distance, which was done with a signal propagation model, described in Section 5.4. After the distance to each AP had been calculated they were used in Equation 12 (seen in Section 10.4), one equation per AP. Solving this set of equations yielded the position.

Pathak et al. proposed a trilateration algorithm that reduces the nonlinear set of equations to a linear set of simultaneous equations, most easily solved using matrices [24]. However, Hereman et al. [44] wrote:

> Coordinates of positions obtained by applying the linear least squares method to the linear system of equations are generally more accurate than the coordinates obtained by solving four equations from the linearized system of equations directly.

The results of Hereman et al. also showed that this method will introduce a relatively large error, and that there are better ways of solving the matrices.

Hereman et al. [44] used a trilateration positioning system to determine the three-dimens-

ional position of equipment in an open pit mine. They explored a number of different techniques for solving the trilateration matrix that calculates the position: linearized system of equations, linear least squares and nonlinear least squares. Their results showed that the nonlinear least squares method gives the most accurate results, providing a position with five feet (or better) accuracy in 91.9% of the measurements. However, we chose to adapt the method which produced the second most accurate results for Hereman et al. , the linear least squares method, displayed in Equation 20, Section 10.4. This because it was easier to implement than the nonlinear least squares method. Refer to Figure 2 for a complete diagram of the chain of algorithms that calculate the position using trilateration.

A potential problem with trilateration is the edge case when only two circles intersect. Qiu et al. solved this by finding the intersection of the first two circles, then drawing a straight line from the intersection to the center of the third circle [45]. The wanted position is then found at the midpoint of the line. This was never implemented in the final version of our system because of time constraints.

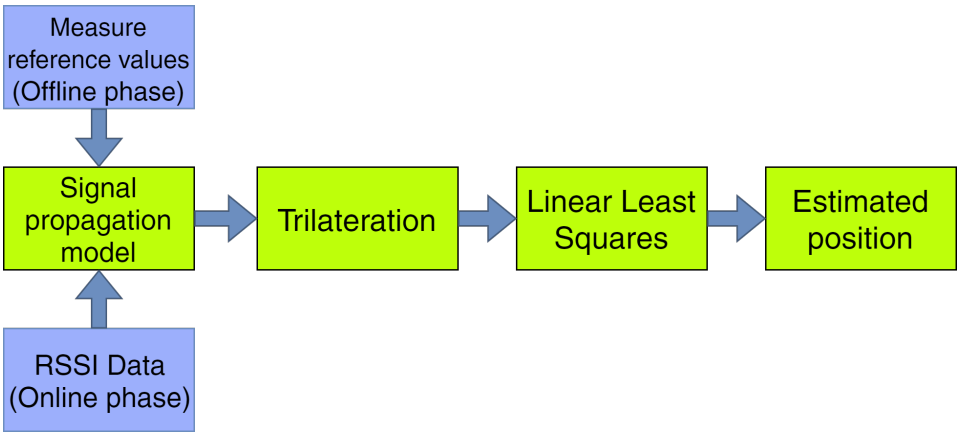

Figure 2: Block diagram of our trilateration positioning method. The blue blocks represent the input to the algorithms, the green blocks represent the algorithms.

5.6 Fingerprinting

In Section 2.5 we briefly explain what fingerprinting is and that positions are determined by querying the database that is built during the offline phase and finding the best matching position. Here we go into detail on different techniques when it comes to finding the best matching position and what techniques we decided to use.

5.6.1 Techniques

Finding the best matching position in a radio map is often done with a variant of the k-Nearest Neighbors (k-NN) algorithm [8, 9, 10, 12]. k-NN algorithms are designed to classify an object to a certain class depending on which its k-nearest neighbors are. An example of a 1-NN algorithm is one of the most basic approaches to fingerprinting: calculating the Euclidean distance between online and offline data to find the best matching location and choosing the location with the lowest Euclidean distance from the online data. Euclidean distance is the straight-line distance between two points (see Section 10.2).

As for generating a radio map, the least complex implementation is what Yiu et al. [8] refer to as traditional fingerprinting. Traditional fingerprinting is a method that relies only on measurements and as Yiu et al. explain:

> For traditional fingerprinting, a measurement campaign has to be carried out for all L locations offline.

This implies that the offline phase in traditional fingerprinting is a labor intensive task as all the locations that the radio map consists of have to be visited and measured. Yiu et al. also identify another problem with traditional fingerprinting: that it performs best when under the unrealistic assumption that data both in online and offline phase are independent and identically distributed. This means that the performance of traditional fingerprint will degrade immediately if the surrounding changes. Examples of such changes are:

- Online and offline data are recorded on different days.
- Person holding the device is not holding the device in exactly the same way all the time.
- Person is not facing the same direction all the time.
- Furniture inside the building is moved.
- APs are removed or added to the building.
- Signal strength of APs fluctuate.

Gansemer et al. [9] mentioned that the human body may have a major influence on the results of what we have described as traditional fingerprinting. Therefore, Gansemer et al. proposed to measure each location in four different headings (0 °, 90 °, 180 °and 270 °) in the offline mode. The adapted Euclidean distances (see Section 10.2) was then calculated between online data and all four headings for each location. The results showed

that the method can deal reasonably well with dynamically changing WiFi environment at the cost of an increased calibration effort.

Quan et al. explored the possibility of only comparing online and offline data within a constrained search space [10]. The idea is that it is physically impossible for a moving object to travel a certain distance within a certain time from its last predicted position. The result showed a better benchmark for the online phase since only locations in a local region were considered. A slight increase in accuracy was also reported since locations that have similar WiFi fingerprints but are physically impossible to reach from the last position are ignored. It should be noted that if this method "got lost" by predicting more than twenty feet away from the actual position, there was little chance of recovery.

Li et al. proposed an improvement to the Min-Max localization algorithm [11]. The Min-Max localization algorithm draws a square around each AP, where the length of the square is relative to the signal strength. These squares intersect, and the middle of the intersecting rectangle is the estimated position. The improvement proposed was a new weight model, a special factor with which each corner of the intersecting rectangle was multiplied. The factor decides how close the estimated position is to the corner: the higher the factor, the closer it is. Their weight model was the best of all the weight models tested through simulation.

Machaj et al. proposed a rank based fingerprinting algorithm [12]. The purpose is to use the rank, i.e. the order, in decreasing similarity, of each point instead of the actual similarity. The position is estimated using the rank in conjunction with a probabilistic model. This is to decrease the need for recalibration of devices, since rank information should be invariant even on devices with different quality of receivers. The results were presented as better than any 1-NN or Weighted k-NN algorithms with the additional advantage of not requiring calibration for better performance.

Yiu et al. provided a comprehensive review on RSSI localization comparing traditional fingerprinting, with a labor intensive offline phase, to two parametric and one non-parametric model for generating the radio map of the building [8]. The parametric models generate a radio map from two different path loss models. The non-parametric model generate a radio map from a statistical model called a Gaussian process (GP) [46]. Results showed that the non-parametric model using GP outperformed traditional fingerprinting at high enough resolutions. The parametric models both fall short of traditional fingerprinting but are not far behind. The advantage of the parametric and non-parametric models is the reduction of labor in the offline phase.

5.6.2 Our choice of techniques

We decided to implement traditional fingerprinting with Adapted Euclidean distance brought up by Gansemer et al. as a similarity comparison method. To get an estimated position out of the similarities we decided to use the Improved Min-Max algorithm brought up by Li et al. The Adapted Euclidean distance algorithm is explained in Section 10.2 and the Improved Min-Max algorithm is explained in Section 10.3. We decided to use these fairly non-complex techniques because that is what we thought was feasible considering the time constraint of the project. In Section 14 our proposed improvements to fingerprinting are discussed.

In both online and offline mode, our system builds a vector for each scan. Each network can be perceived as a dimension where the signal strength from that network is the length of the vector in that dimension. To ensure that the Adapted Euclidean distance algorithm compares correct networks against each other, the vectors need to be built with identical format. In our case this means that if A and B are the two vectors to compare, the RSSI value in A_i needs to correspond to the same network as the RSSI value in B_i.

The complete chain of algorithms to calculate a position using our method of fingerprinting is visualized in Figure 3.

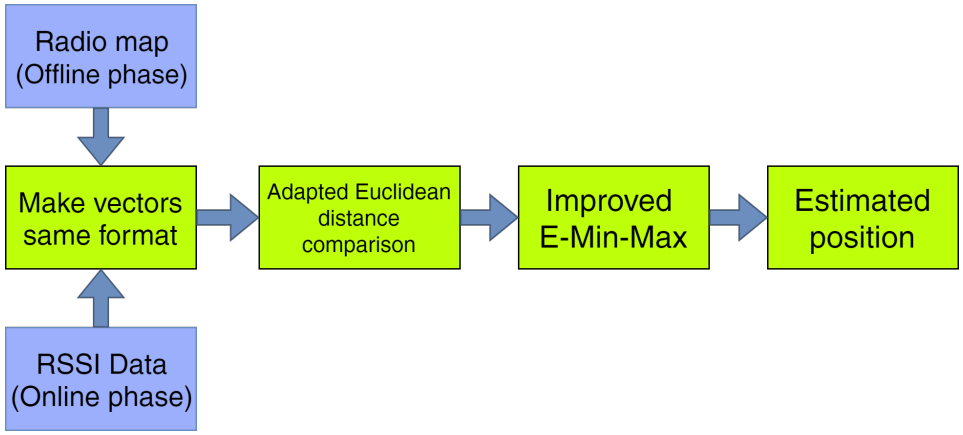

Figure 3: Block diagram of our fingerprint positioning method. The blue blocks represent the input to the algorithms, the green blocks represent the algorithms.

6 System structure

In this section we explain the different parts of the constructed system and how they relate to each other. Figure 4 shows how our system interacts with Pythagoras' preexisting system. Each block in the picture describes a part of the system where entities inside that block are connected locally. Solid arrow lines describe data flow between the different parts of the system, which in our case is done with HTTP calls through RESTful web services. Dashed arrow lines describe data flow between entities connected on a local level. The dotted arrow line between SpaceManager and Pythagoras' database is communication done with the MySQL protocol. Solid lines represent interactions between the user and the system.

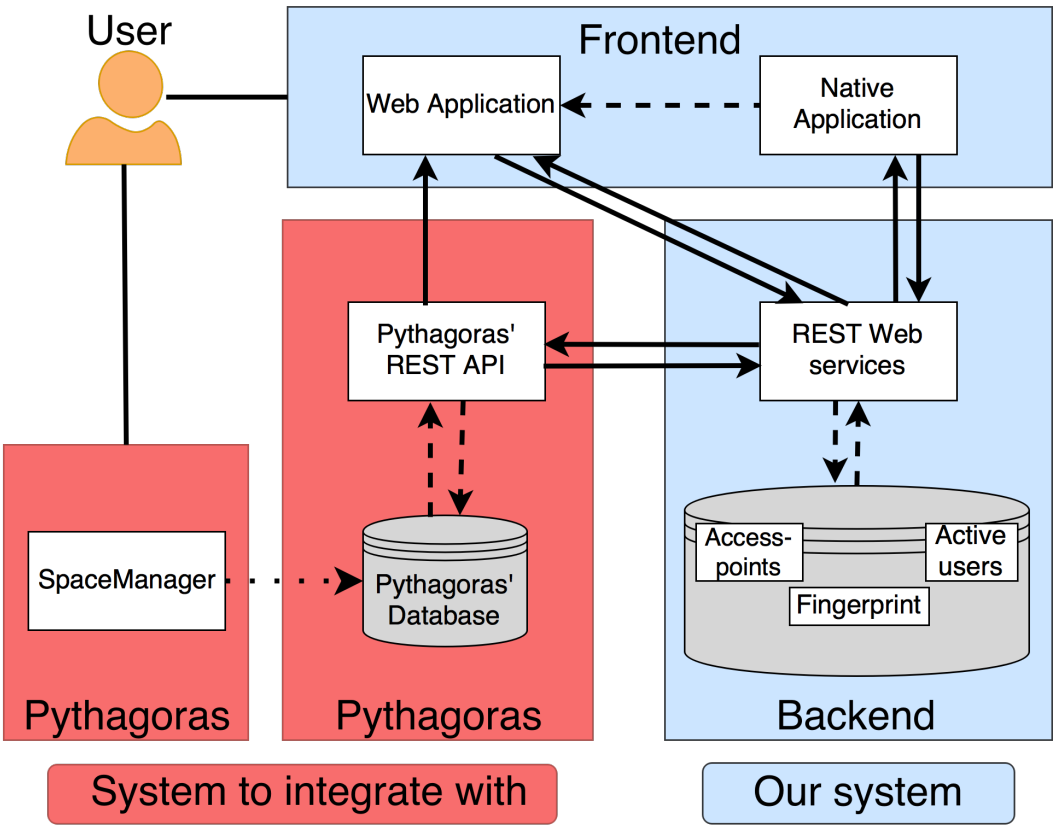

Figure 4: Block diagram of how our system integrates with Pythagoras' system.

The frontend is comprised of two parts: a web application and a native application. The native application is Android code running on the device. The web application is being run in a web container, a lightweight web browser, visible inside the native application.

When the frontend starts on a device, the native application queries the backend for a user ID, which is forwarded to the web application. The web application queries the backend at regular intervals for markers to display, and a map ID that indicates on which map they are to be displayed. The map ID is sent to Pythagoras' API which responds with the map corresponding to that ID. When navigating the frontend, the native application notifies the web application from which web service it should fetch markers, which is then displayed to the user. The native application displays all buttons and handles the logic of gathering RSSI data. All RSSI data gathered is sent to the backend, formatted as JSON objects, for processing.

When the backend processes the data it also stores the result. Where it is stored depends on which web service was called. When gathering fingerprinting data, it will be stored in a fingerprint database. When a position is calculated, it is stored, alongside the user ID, in the active user database table. If an AP is to be calibrated, the measurements will be stored in the local access point database table, as well as sent to Pythagoras' server using their API where it is stored in the relevant database table.

A user will interact with either the frontend on a smartphone device or with SpaceManager on a computer running Microsoft Windows. In SpaceManager, the user adds or removes APs to or from the digital representation of a building using a visual interface. The usage of the smartphone application is described in Section 8.

6.1 Alternative solutions

An alternative solution would be to house the positioning functionality in the native application. The advantage of this would be less strain on the server, since the calculations would not need to be done on the server. It would also result in less delay for the client, from measuring RSSI data to receiving their position.

Disadvantages of migrating the positioning functionality are:

- An increase in network traffic between the client and server since the server would need to send all RSSI fingerprinting data to the client.
- Future expansions such as using a building's magnetic signature and the magnetometer of smartphones would result in even more network data.
- Migrating to the native application would mean that each time the system receives an update all clients would need to update their application. This is not a problem if the functionality is, as it is now, implemented in the backend.

7 Requirements and evaluation methods

The requirements for this project were decided in collaboration with Pythagoras. Their wish was for the system to be able to position a device in the correct room. Since a room can be of arbitrary size, a maximum allowed error was chosen. For the system to be considered successful at positioning, the calculated position was to be no more than four meters from the actual position 95% of the time. In addition to that we also attempted to answer the questions asked in Section 3: How can the system be easily configured by a customer? How accurate can a user configured WiFi-only positioning system be?

An experiment campaign was carried out to evaluate the requirement of four meters and to answer the question: How accurate can a user configured WiFi-only positioning system be? Experiments were conducted in Pythagoras' office on floor six in Münchenbryggeriet, Stockholm, Sweden. The system was tested using four different techniques: trilateration[3], fingerprinting[4], evenly weighted results from trilateration and fingerprinting[5], and closest fingerprinting location[6]. A total of 30 points, as shown in Figure 5, was scattered throughout the building. They were placed so that there would be at least one point to scan in each room, and in larger rooms the testing points were placed pseudo-randomly in the room, approximately four meters apart from each other. At each testing point it was determined whether the calculated position was within two, three or four meters from the actual position, and whether it was considered to be in the same room or not. Experiments were made on two different phones: Sony Xperia Z5 Compact and Samsung Galaxy S6. The 30 points were measured four times using an old[7] radio map and once using a brand new radio map. The old radio map was generated using a Sony Xperia Z5 and the brand new radio map was generated using a Samsung Galaxy S6. Both radio maps were made with a two meter distance between each measurement.

[3] See section 5.5 for explanation

[4] See section 5.6.2 for explanation

[5] Midpoint of the trilateration position and the fingerprint position

[6] Positioning done by what is described as a basic fingerprinting algorithm in Section 5.6.1

[7] The old radio map was eight to ten days old. Measurement was done over a span of three days.

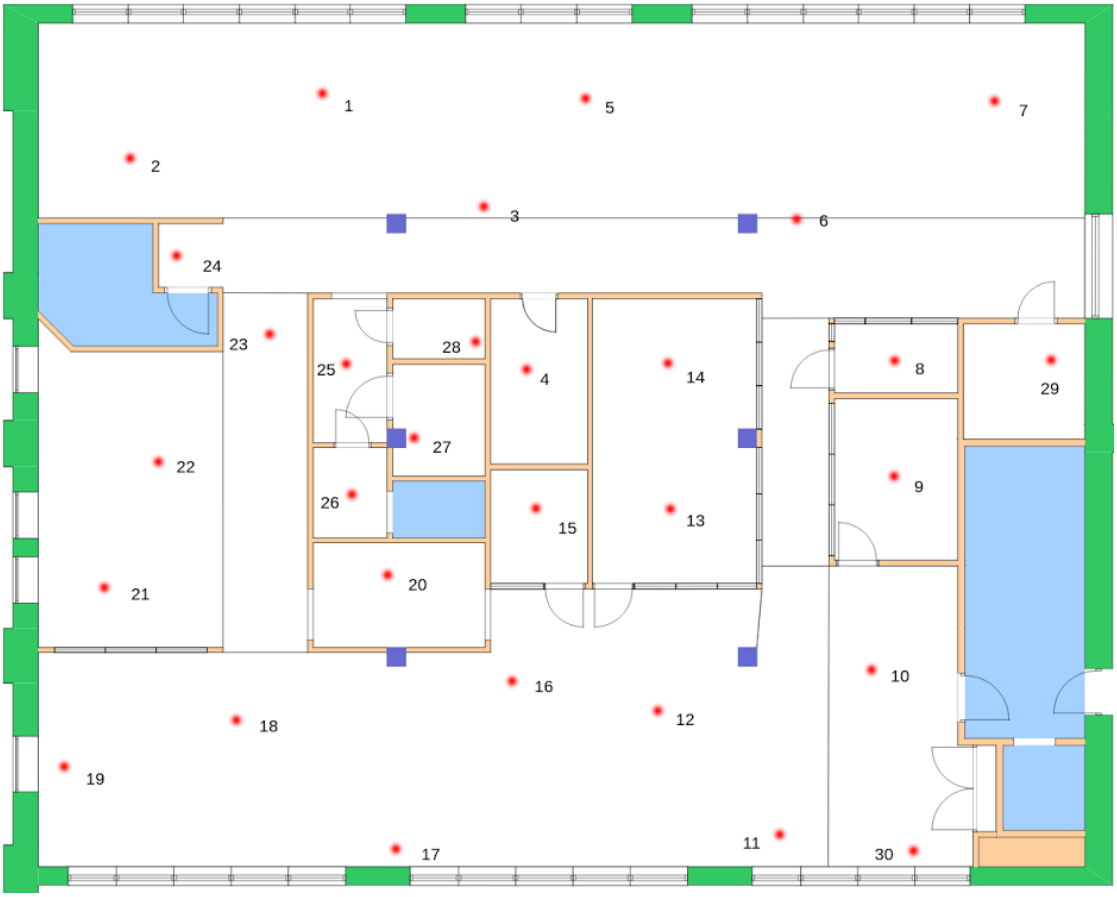

Figure 5: A map of Pythagoras' office floor. The red dots are the test points used when evaluating the system, the blue squares are pillars, the blue rooms are rooms not belonging to Pythagoras.

No evaluation was made to answer the question: How can the system be easily configured by a customer? Instead, the system was developed with the assumption that any customer can visit points displayed on a map and tap a button. We consider the configuration process (as described in Section 9) to be trivial enough that no particular evaluation is needed to confirm that a customer can perform it given a brief introduction of less than a minute.

8 The product

The final product was built to have only the three main functionalities necessary for testing the system, viz.: positioning, AP calibration and fingerprinting calibration. This section intends to give a more detailed explanation of how these three modes work.

Live mode In live mode, the native Android part of the application scans for RSSI data at regular intervals and sends it, together with a unique user ID (represented as a string of twelve random lowercase letters and numbers), to the server. The web application queries the server at regular intervals for its and other active users position and displays it on a map as seen in Figure 6. The position received consists of the two-dimensional Euclidean coordinates, as well as a unique floor ID (represented as an integer). The blue circle with a white stroke and an outer blue circle indicates where the device is located. The outer blue circle is set to four meter. Red circles with a black stroke indicated other devices position in the system. A user can at any time choose to configure the buildings by tapping one of the two calibration button located at the bottom of the screen.

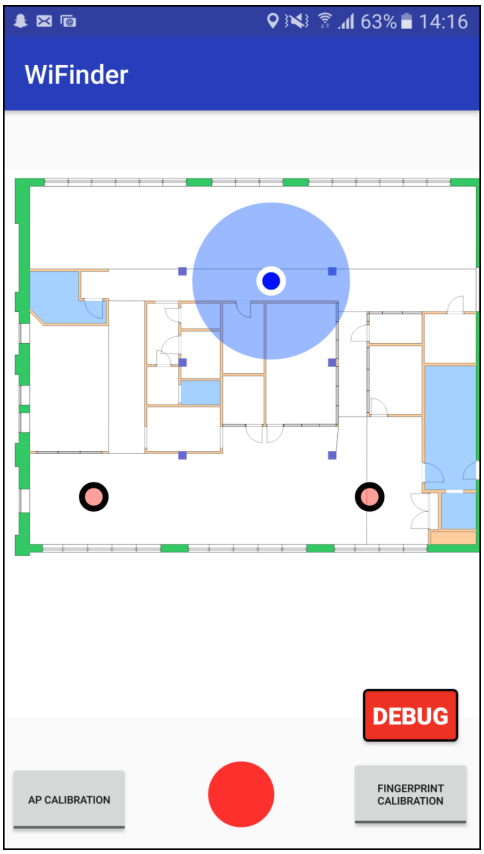

Figure 6: A screenshot of how the product looks in live mode. The smaller blue circle is the position of the device. The larger blue circle is a circle with a radius of four meters, the group's personal accuracy goal (mentioned in Section 3). The small red circles are the positions of other devices currently using the application. The big red circle is an indicator that the application is in live mode.

AP calibration mode The AP calibration mode is meant to measure a reference value of an AP at a distance of one meter, which is used for trilateration. In this mode the native part queries the server for an AP to visit. When successfully retrieved, the web application will display a dot at the position of the AP with instructions of where to scan. Upon a completed scan, all information of the scan is sent to the server. The server determines which AP has the strongest signal and assumes it to be the AP to be calibrated. That AP's Basic service set identifier (BSSID), which is the unique identifier of a network, and RSSI value are then stored for future use. The RSSI value will be used as a reference value for trilateration. A walkthrough of how AP calibration mode is used can be found in Section 9.

Fingerprinting mode When fingerprinting mode is activated, the native part of the application requests a list of points from the server to visit. When fetched, these are sent to the web application, which in turn displays them as dots on the map. The next point in the list to be visited is seen in red. When the scan has been completed, an object containing information about the ID of all networks and their respective RSSI values is sent to the server. When the native part of the application gets an acknowledgement that the data has been received by the server, it sends a message to the web application, instructing it to mark the scanned point in green and the next point in red. This is repeated until all gray markers have been visited and scanned. A walkthrough of how fingerprinting mode is used can be found in Section 9.

9 User configurability

This section intends to show how the system can be configured by a customer. This is split into two parts: first, how APs can be added to the digital representation using the facility management software SpaceManager, and second, how the Android application can be used to improve future positioning by calibrating APs and fingerprint positions.

SpaceManager Pythagoras provides a facility management software, SpaceManager. SpaceManager allows a customer to add, remove and modify components such as furniture, plants and doors in the digital representation of a building. We have created a component which represents an AP. This component can be added to any building using SpaceManager as shown in Figure 7.

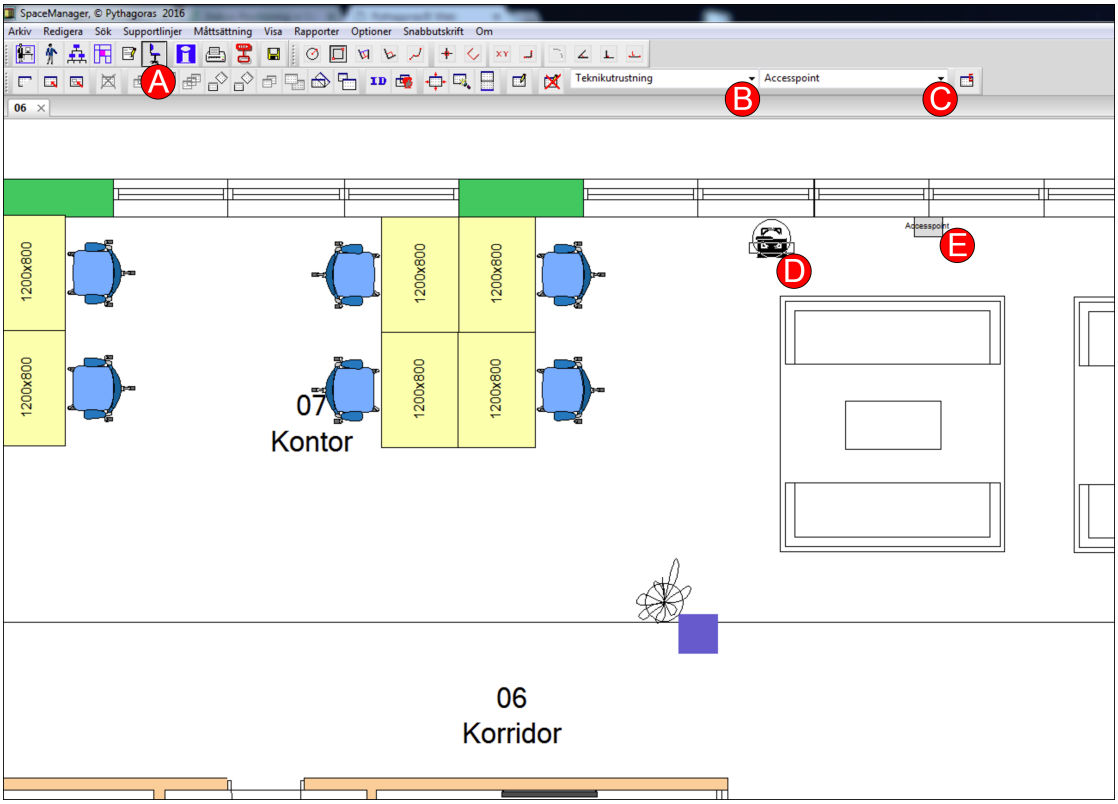

Figure 7: A screenshot of SpaceManager. A: The component button. B: Drop down menu for choosing component category. C: Dropdown menu for selecting a component. D: The AP being placed by the current user. E: An AP which is already part of the digital representation.

The steps required to add an AP to the digital representation of a building:

1. Click on the component button (A).
2. Choose the category ”Teknikutrustning” which translates to ”Technical equipment” (B).
3. Choose the component ”Accesspoint” (C).
4. Place the AP by moving the cursor and left clicking (D).

Calibrating an AP When an AP has been successfully added in SpaceManager, the AP needs to be given a BSSID and a reference value. This is done in the Android application as shown in Figure 8.

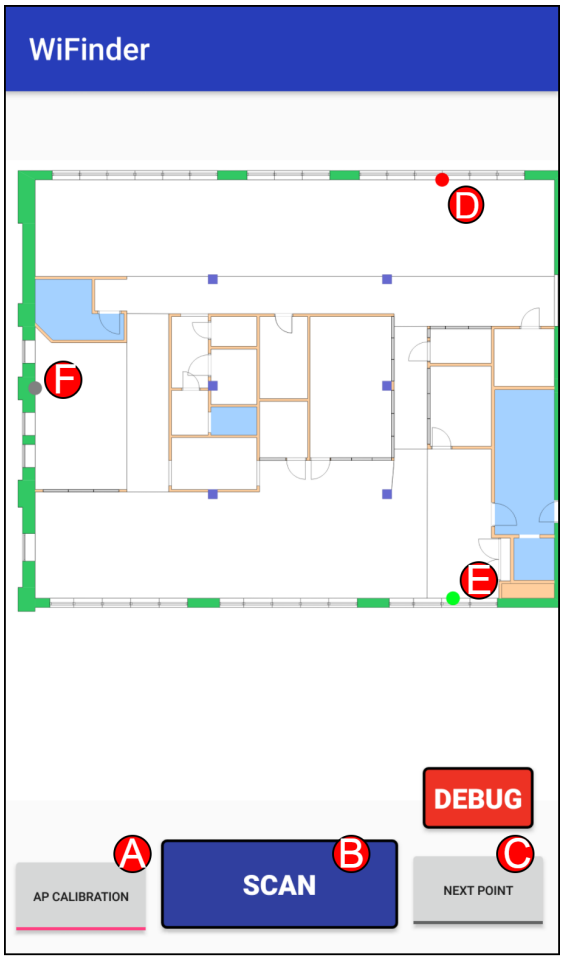

Figure 8: A screenshot of AP calibration mode in the Android application. A: The button to activate AP calibration mode. B: The button to start a scan. C: The button to skip a scan point. D The next point to be scanned in red. E: A previously scanned point in green. F: A point to be scanned in the future in grey.

These are the steps a customer needs to execute to calibrate an AP in their building:

1. Tap the “AP CALIBRATION” button to go into AP calibration mode (A).
2. Go to the red circle indicating the next AP to be calibrated (D) or tap ”NEXT POINT” to skip a point (C).
3. Tap ”SCAN” and wait until the circle turns green (B).

4. Repeat step 2-3 until all points one wishes to scan are green.

Calibrating a fingerprint position Figure 9 shows how a user can calibrate the fingerprint of a building.

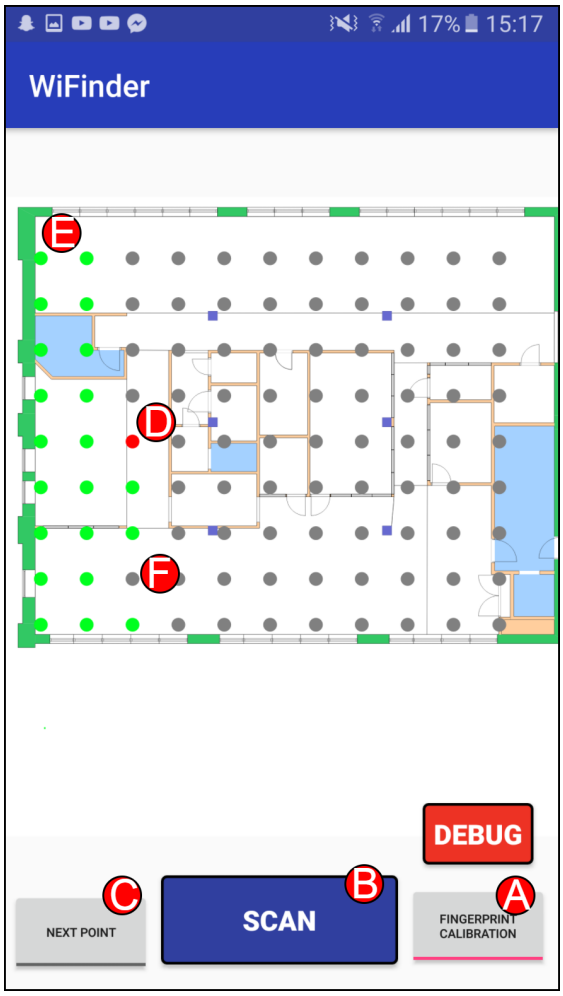

Figure 9: A screenshot of fingerprint calibration mode in the Android application. A: The button to activate fingerprint calibration mode. B: The button to start a scan. C: The button to skip a scan point. D: The next point to be scanned in red. E: A previously scanned point in green. F: A point which has not been scanned in gray.

These are the steps a customer needs to execute to calibrate the fingerprint of a building:

1. Tap the “FINGERPRINT CALIBRATION” button to go into fingerprint calibration mode (A).

2. Go to the red circle (D) , indicating the point to be scanned or tap "NEXT POINT" to skip the current point (C).
3. Tap the "SCAN" button and wait for the red circle to turn green (B).
4. Repeat step 2-3 until all points one wishes to scan are green.

10 Positioning algorithms

In order to provide accurate positions to the mobile application, several algorithms and models for positioning were used. This section intends to give a thorough explanation of them.

10.1 Signal propagation

As mentioned in Section 5.4, we made use of the path loss model expressed by Ogue-jiofor et al. as it was clear and concise [41]. We also used the model proposed by Akl et al. because it was improved upon by accounting for environmental factors using a Gaussian distributed random variable [42]. Our final model was expressed in the following way:

$$RSSI[dBm] = -10n \log_{10}(d) + A[dBm] + X_\sigma[dBm] \tag{1}$$

Where d is the modeled distance (in meters) from the current position to the AP, $RSSI$ is the value measured at the current position, A is the reference value measured one meter from the AP, X_σ is a zero-mean Gaussian distributed random variable with a standard deviation of σ and n is the PLE that is dependent on the current environment and has to be derived empirically.

We measured the reference values individually for each BSSID and stored them in our database. They ranged from -41 to -55. Based on trial-and-error testing, measuring RSSI at known distances, at Pythagoras' office we concluded that σ should be equal to 3 dBm and that the PLE should be equal to 3.

10.2 Adapted Euclidean distance

As mentioned in Section 5.6.2, we decided to do similarity comparisons using adapted Euclidean distance. Adapted Euclidean distance is the Euclidean distance divided by the number of dimensions squared [9]. In our case, the number of dimension was the number of networks to be compared. Euclidean distance is the straight-line distance between two points. In two dimensions, between point a and b, this was calculated by solving

$$d = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} \tag{2}$$

We can calculate this for any arbitrary number of dimensions. If A and B are two vectors to compare, then the size of the vectors, n, is the number of dimensions and each value A_i and B_i is the corresponding coordinate of the vector in dimension i. The Euclidean distance is calculated by solving

$$d = \sqrt{\sum_{i=1}^{n} (A_i - B_i)^2} \tag{3}$$

The adapted Euclidean distance is calculated by solving

$$d = \sqrt{\frac{\sum_{i=1}^{n} (A_i - B_i)^2}{n}} \tag{4}$$

We used adapted Euclidean distance instead of normal Euclidean distance since we could not guarantee that every vector has the same number of dimensions. The adapted Euclidean distance prevented a vector comparison from erroneously accumulating a longer distance due only to the fact that some vectors might have had more dimensions than others.

10.3 Improved Extended Min-Max localization algorithm

Once similarities were obtained from calculating the adapted Euclidean distance we calculated the vertices p_1 through p_4 (see Figure 10 and Equations 5 through 8) using the

known coordinates of the fingerprinting positions and their adapted Euclidean distances d_i. The result was the estimated position shown in Figure 10, but also shown in the figure is that the estimated position was probably not perfect. However, we improved this by utilizing the improved Extended Min-Max (E-Min-Max) localization algorithm proposed by Li et al. We weighed the coordinates of the vertices with the weight that Li et al. found to cause the smallest localization error [11], shown in Equation 9.

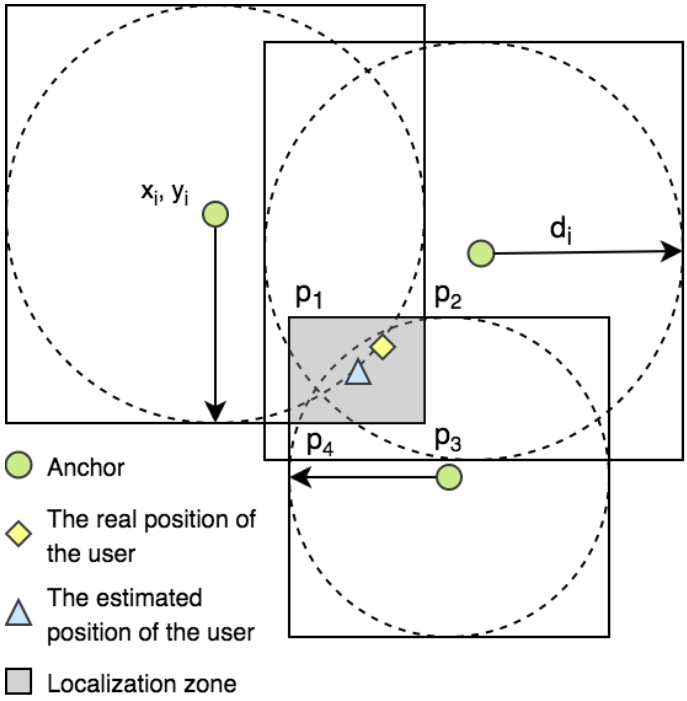

Figure 10: Visualization of the Min-Max algorithm.

The algorithm begins by calculating the coordinates of the vertices p_1 to p_4, seen in Figure 10, using the following equations described in Li et al. [11]:

$$p_1 = [max(x_i - d_i), min(y_i + d_i)] \tag{5}$$

$$p_2 = [min(x_i + d_i), min(y_i + d_i)] \tag{6}$$

$$p_3 = [min(x_i + d_i), max(y_i - d_i)] \tag{7}$$

$$p_4 = [max(x_i - d_i), max(y_i - d_i)] \tag{8}$$

After the vertices were calculated the position was estimated to be in the middle of the location zone, as seen in Figure 10. It is improbable that the position is right in the middle of the localization zone, which is why we weighed the coordinates of the vertices with a weight model to get the estimated position closer to the real position. This is the E-Min-Max algorithm. The weight model we used was proposed by Li et al. in their improved E-Min-Max, and it got the smallest localization error out of all the weight models tested in their simulation [11]. It is modeled like this

$$W(j) = \frac{1}{\sum_{i=1}^{n}(M_{i,j} - d_i)^2} \tag{9}$$

Where $W(j)$ is the weight for vertex j, i is the AP, $M_{i,j}$ is the Manhattan distance between AP i and vertex j, and d_i is the Euclidean distance from AP i to the position. The Manhattan distance is the distance between two points in a grid, based on the horizontal and/or vertical distance instead of the diagonal distance. It is calculated as follows

$$M_{i,j} = |x_i - x_j| + |y_i - y_j| \tag{10}$$

The final estimated position was calculated by applying the weight model to the coordinates of each vertex. The final equation is expressed this way

$$(x, y) = \left(\frac{\sum_{j=1}^{4} W(j) \cdot x_j}{\sum_{j=1}^{4} W(j)}, \frac{\sum_{j=1}^{4} W(j) \cdot y_j}{\sum_{j=1}^{4} W(j)} \right) \tag{11}$$

10.4 Trilateration

Trilateration is explained generally in Section 2.2. As mentioned in Section 5.5 there is a distance equation for each AP_i, namely Pythagoras' theorem.

$$r_i^2 = (x - x_i)^2 + (y - y_i)^2 \tag{12}$$

Where x, y are the coordinates to be found, and x_i, y_i are the coordinates of AP_i. To solve this, we started with linearizing the system by adding and subtracting x_j and y_j,

where x_j, y_j are the coordinates of AP_j.

$$r_i^2 = (x - x_j + x_j - x_i)^2 + (y - y_j + y_j - y_i)^2 \tag{13}$$

As shown in Hereman et al. [44], this leads to

$$b_{ij} = \frac{1}{2}(r_j^2 - r_i^2 + d_{ij}^2) \tag{14}$$

where

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{15}$$

is the distance between AP_i and AP_j. Since it does not matter which AP is used to linearize the system [44], the first one was arbitrarily chosen ($j = 1$). For three APs in two-dimensional space the linear system is

$$b_{21} = \frac{1}{2}(r_1^2 - r_2^2 + d_{21}^2) \tag{16}$$

$$b_{31} = \frac{1}{2}(r_1^2 - r_3^2 + d_{31}^2) \tag{17}$$

which can be represented in matrix form as

$$\mathbf{A}\vec{x} = \vec{b} \tag{18}$$

where

$$\mathbf{A} = \begin{pmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{pmatrix}, \quad \vec{x} = \begin{pmatrix} x - x_1 \\ y - y_1 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} b_{21} \\ b_{31} \end{pmatrix} \tag{19}$$

This linearized system could have be solved directly, but as mentioned in Section 5.5 it would have been suboptimal, which is why we used the linear least squares method instead.

$$\mathbf{A}^T\mathbf{A}\vec{x} = \mathbf{A}^T\vec{b} \tag{20}$$

Solve for x.

$$\vec{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\vec{b} \tag{21}$$

11 Evaluation results

This section contains the results from the evaluation methods described in Section 7. Each table shows the percentage of measurements which resulted in an estimated position within two, three and four meters of the actual position. Percentage of measurements which resulted in an estimated position inside the room in which the measurement was made are presented in the "room" column. For fingerprinting, results are also shown when using an old and a new radio map, where the old map was created using the Sony smartphone and the new was made using the Samsung device. The old radio map was eight to ten days old as measurement were carried out over a three day period. Results are presented in Tables 1 through 4.

Trilateration				
Phone	2m	3m	4m	room
Samsung	20.4%	32.9 %	50.0%	30.0 %
Sony	18.3%	29.6 %	34.4 %	20.0 %
Average	19.4 %	31.3 %	42.2 %	25.0 %

Table 1: The results of trilateration.

Improved E-Min-Max fingerprinting								
	2m		3m		4m		room	
Phone	old	new	old	new	old	new	old	new
Samsung	36.1%	60.0%	66.7%	80.0%	85.0%	90.0%	55.6%	63.0%
Sony	31.7%	20.0%	52.8%	50.0%	65.0%	60.0%	41.7%	60.0%
Average	33.9%	40.0%	59.7%	65.0%	75.0%	75.0%	48.6%	61.7%

Table 2: The results of our fingerprinting algorithm when positioning against an old and a brand new radio map.

Closest fingerprinting location								
	2m		3m		4m		room	
Phone	old	new	old	new	old	new	old	new
Samsung	23.9%	50.0%	43.3%	66.7%	53.3%	80.0%	37.8%	63.3%
Sony	31.7%	23.3%	53.3%	50.0%	60.0%	63.3%	45.0%	53.3%
Average	27.8%	36.7%	48.3%	58.3%	56.7%	71.7%	41.4%	58.3%

Table 3: The results of the basic fingerprinting algorithm when positioning against an old and a brand new radio map.

Evenly weighted trilateration and E-Min-Max fingerprinting								
	2m		3m		4m		room	
Phone	old	new	old	new	old	new	old	new
Samsung	34.4%	33.3%	65.0%	53.3%	86.7%	76.7%	49.4%	50.0%
Sony	32.2%	13.3%	50.0%	43.3%	61.7%	36.7%	41.1%	40.0%
Average	33.3%	23.3%	57.5%	48.3%	74.2%	56.7%	45.3%	45.0%

Table 4: The results of weighted trilateration and fingerprinting when positioning against an old and a brand new radio map.

12 Results and discussion

As seen in Section 11, none of the positioning techniques resulted in achieving the goal of an error under four meter 95% of the time. The closest result to four meter accuracy 95% of the time was achieved by fingerprinting using a brand new radio map and using the same device for positioning as was used for generating the radio map. Under those circumstances, a four meter accuracy could be established 90% of the time. This was

done using the improved E-Min-Max algorithm, which generated the best result out of all techniques investigated if average accuracies were compared.

Overall, fingerprinting performed best when using the same device for positioning and calibration. This is seen in Table 2 and 3 where the Sony phone was used to generate the old radio map and the Samsung phone was used to generate the new radio map. From the tables we can see that the Sony phone had better accuracy on the old radio map compared to the new, while the Samsung phone performed better on the new radio map compared to the old.

When averaging the results from both phones, we could see that a new radio map resulted in less error compared to an older radio map. This is because the radio fingerprint of a building changes over time as furniture and APs are moved, removed or added.

From all four techniques, we could see that the Samsung phone had better accuracy. The reason for this is not completely clear but we think it could be attributed to the Samsung device having receivers that are not as easily blocked as the Sony. It might also be related to the quality of antennas or WiFi chips.

Results from trilateration were generally worse than fingerprinting. Our current signal propagation model was a rough estimate, as the signal does not uniformly propagate from the AP. Our representation of it as a circle was therefore far from perfect. Even with a Gaussian random variable, a small change in position could result in vastly different estimated position, particularly if there was a wall separating the two positions. The poor accuracy is also due to the fact that Pythagoras' office did not have complete coverage from their own APs. This resulted in some scans only detecting two known APs, which would lead to an estimated position that was several meters from the actual position.

The evenly weighted results between trilateration and fingerprinting, Table 4, were generally worse than fingerprinting using improved E-Min-max, apart from a few exceptions. Those exceptions occurred when positioning was done against an older radio map. During testing, we noticed that the evenly weighted result would either help position by compensating for fingerprinting error, or spoil the positioning completely when results from the trilateration was poor. The reason for the evenly weighted result performing better on an old radio map compared to a new is unknown to the group.

Our system had poor results for positioning a device inside a room. This could be attributed to some rooms being particular hard to position a device inside. Figure 11 shows which locations had poor room positioning. These were areas typically identified as small rooms.

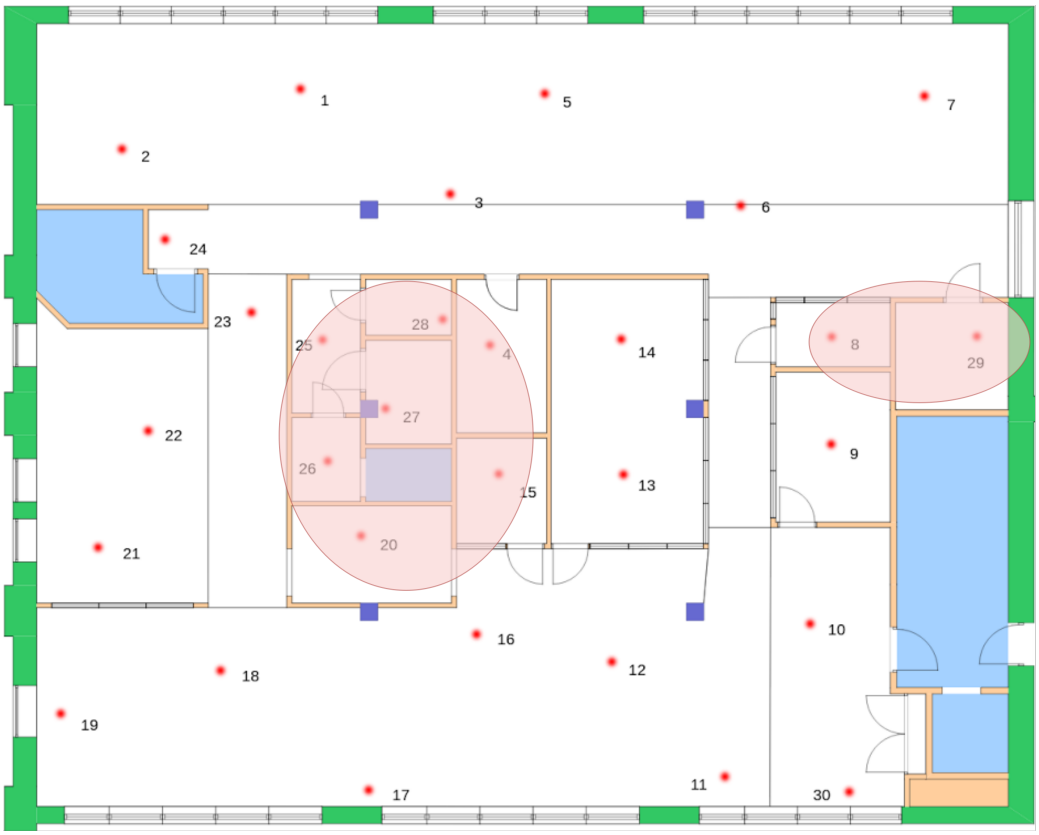

Figure 11: Areas suffering from poor (10% or less) room positioning rate.

The shaded areas in Figure 11 are points where the estimated position was inside the room of measurement less than 10% of the scans. Considering the size of these rooms, it is not surprising that the algorithms had a hard time estimating a position inside the rooms since an error as small as one meter could result in the estimated position being outside the room of measuring.

Although it was not part of the requirements, the time it took to create a radio map was a factor which was taken into consideration, as customers want to spend as little time as possible configuring the system. During this project, creating a full fingerprint of the 414 m^2 office took around 45 minutes. When consulting with Pythagoras, it was concluded that this was acceptable since, at this rate, a 4,416 m^2 area could be scanned in a working day of eight hours. This corresponds to approximately a ten story version of Pythagoras' office. While this might be acceptable as a "one time" scan, it should be noted that since radio maps degrades over time, fingerprinting in this manner, might prove too cumbersome to maintain in the long run.

It can be difficult to relate to other works dealing with RSSI positioning. This is because buildings and WiFi infrastructure used during testing can vary vastly between projects and tests. Nonetheless, comparing to other projects can still give an indication of whether the algorithms used were implemented in a correct way. Pathak et al. managed to get a trilateration accuracy of around two meters [24], clearly better than this project achieved. However, no position tested was further than five meters from an AP and only six points were tested. The rank based algorithm proposed by Machaj et al. generated a mean error of approximately five meters [12]. The best case results presented here in Section 12 has similar results. However, Machaj et al. managed to get consistent errors, independent of the device used for positioning [12]. This is something the system constructed in this project completely fails to do, as the error fluctuates quite heavily depending on which device was used, as seen in e.g. Table 2.

13 Conclusions

We have shown that a user configurable indoor positioning system using WiFi can be built with the requirement that there exists a digital representation of the buildings to be included in the system. However, the positioning results shows that our current implementation of WiFi positioning is not accurate enough to allow for room scaled positioning. Fingerprinting using the improved Extended Min-Max algorithm was superior to trilateration in its positioning accuracy, but manually fingerprinting a floor requires a non-negligible time investment.

The system can be used as a core for coarse navigation. In industrial facilities such as factories, exact, room specific navigation might not be necessary, but instead positioning a user in a specific department may be sufficient for usability. In such cases, the current positioning accuracy would be acceptable.

Configuring the system is the most critical part for accuracy. However, the configuration could be done by any person given some experience using a smartphone. This results in a system which could be configured with low workload by any employee regardless of technical background, given that building in question is already represented in Pythagoras' facility management system.

14 Future work

For the system to be viable for commercial deployment, several areas need to be improved. The following are suggestions for areas of focus in the event that the system is developed further.

Fingerprinting Replacing the method for fingerprinting would be of importance since the offline phase currently is quite labor intensive. One way to do this is to implement an algorithm for simultaneous localization and mapping [47], i.e. make the mapping process more automatic. Reducing the time it takes to map a building would most likely increase potential customers' interest since there is less work to be done for a successful installation in a building. Another potential way to reduce the labor of the offline phase is to move away from traditional fingerprinting like Yiu et al. and use a Gaussian process model to generate the radio map [8]. While algorithms used for positioning can be improved [10, 12], we consider it more important to improve the effectiveness of the offline phase. This is because we regard the biggest flaw of traditional fingerprinting to be that it is too cumbersome to realistically maintain.

Signal propagation The signal propagation model could be improved by better tailoring it to the environment.

- Adapting a more intricate signal propagation equation that takes into account receiver and transmitter antenna gain as well as signal wave length, e.g. the Hata-Okumara model [7].
- Measuring the reference value multiple times and calculating an average.
- Calculating the PLE [7, 41, 42, 43] and the standard deviation of the Gaussian random variable [42, 43].
- Using unique variables for each AP [42].
- Applying different PLEs depending on the signal strength [7].

Trilateration In this project, the digital representation of a building, provided by Pythagoras, was only used for fetching the location of APs. By using knowledge of the environment from the digital representation, i.e. walls and furniture placement, the accuracy could be improved. Taking this information into account when estimating a distance from an AP would result in a more accurate model of signal propagation. Solving the system of distance equations (displayed in Equation 12 in Section 10.4) with the nonlinear least squares method instead of the linear least squares method would also

improve accuracy [44]. Handling the edge case when only two circles intersect would result in better performance in suboptimal environments [45].

User interface Even though it was considered outside the scope of this project, the user interface could be greatly improved. Currently, for a user to succeed with mapping a building, a short explanation of the factors which play into a good fingerprint is needed. The same is true for calibrating APs. A better interface would be a way to ease the process of calibrating the system.

References

[1] "Apple: Introduces the New iPhone 3G," accessed: 2017-05-09. [Online]. Available: https://web.archive.org/web/20170509112741/https://www.apple.com/pr/library/2008/06/09Apple-Introduces-the-New-iPhone-3G.html

[2] K. Marsal, Dec 2008, accessed: 2017-05-14. [Online]. Available: http://web.archive.org/web/20170514135653/http://appleinsider.com/articles/08/12/02/iphone_single_handedly_driving_smartphone_growth.html

[3] "GPS.gov: GPS Accuracy," accessed: 2017-03-30. [Online]. Available: https://web.archive.org/web/20170330140954/http://www.gps.gov/systems/gps/performance/accuracy/

[4] "Pythagoras," accessed: 2017-04-06. [Online]. Available: http://web.archive.org/web/20170406103544/http://www.pythagoras.se/en/index.html

[5] *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007): IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access C*. S.l.: IEEE, 2012.

[6] "Android: RSSI level," accessed: 2017-05-09. [Online]. Available: https://developer.android.com/reference/android/net/wifi/ScanResult.html#level

[7] A. Bose and C. H. Foh, "A practical path loss model for indoor wifi positioning enhancement," in *Information, Communications & Signal Processing, 2007 6th International Conference on*. IEEE, 2007, pp. 1–5.

[8] S. Yiu, M. Dashti, H. Claussen, and F. Perez-Cruz, "Wireless RSSI fingerprinting localization," *Signal Processing*, vol. 131, pp. 235 – 244, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168416301566

[9] S. Gansemer, U. Großmann, and S. Hakobyan, "RSSI-based Euclidean Distance algorithm for indoor positioning adapted for the use in dynamically changing WLAN environments and multi-level buildings," in *2010 International Conference on Indoor Positioning and Indoor Navigation*, Sept 2010, pp. 1–6.

[10] M. Quan, E. Navarro, and B. Peuker, "Wi-fi localization using rssi fingerprinting," 2010.

[11] R. R. Li, L. Li, and X. H. Li, "An improved localization algorithm in wireless sensor networks," in *Advanced Materials Research*, vol. 1049. Trans Tech Publ, 2014, pp. 2144–2148.

[12] J. Machaj, P. Brida, and R. Piché, "Rank based fingerprinting algorithm for indoor positioning," in *2011 International Conference on Indoor Positioning and Indoor Navigation*, Sept 2011, pp. 1–6.

[13] "GPS.gov: GPS Overview," accessed: 2017-05-09. [Online]. Available: http://web.archive.org/web/20170509122922/http://www.gps.gov/systems/gps/

[14] N. C. Gupta and I. Books24x7, *Inside Bluetooth Low Energy*. Boston: Artech House, 2013.

[15] "IDC: Smartphone OS Market Share," accessed: 2017-03-30. [Online]. Available: https://web.archive.org/web/20170330133603/http://www.idc.com/promo/smartphone-market-share/os

[16] "Android: Android Versions," accessed: 2017-05-09. [Online]. Available: https://developer.android.com/about/dashboards/index.html

[17] "Estimote," accessed: 2017-04-06. [Online]. Available: http://web.archive.org/web/20170406103636/http://estimote.com/

[18] "Senion: FAQ," accessed: 2017-05-10. [Online]. Available: https://senion.com/faq/

[19] "MazeMap," accessed: 2017-04-06. [Online]. Available: http://web.archive.org/web/20170406104722/https://www.mazemap.com/

[20] "Steerpath: Features," accessed: 2017-05-10. [Online]. Available: http://www.steerpath.com/features-v3/#anchor-feature-positioning

[21] "infsoft: Navigation using Beacons," accessed: 2017-05-10. [Online]. Available: https://www.infsoft.com/technology/sensors/bluetooth-low-energy-beacons

[22] "Polestar: Indoor Positioning," accessed: 2017-05-10. [Online]. Available: http://www.polestar.eu/en/products/indoor-positioning.html

[23] "iOS Developer Program Information," accessed: 2017-04-06. [Online]. Available: http://web.archive.org/web/20170406103510/https://developer.apple.com/programs/ios/information/iOS_Program_Information_4_3_15.pdf

[24] O. Pathak, P. Palaskar, R. Palkar, and M. Tawari, "Wi-Fi Indoor Positioning System Based on RSSI Measurements from Wi-Fi Access Points—A Trilateration Approach," *International Journal of Scientific and Engineering Research*, vol. 5, pp. 1234–1238, 2014.

[25] "CampusGuiden," accessed: 2017-04-06. [Online]. Available: http://web.archive.org/web/20170406104645/http://www.campusguiden.no/

[26] "Accuware," accessed: 2017-05-11. [Online]. Available: http://web.archive.org/web/20170511084748/https://www.accuware.com/

[27] "Accuware: Accuracy," accessed: 2017-05-11. [Online]. Available: http://web.archive.org/web/20170511084004/https://www.accuware.com/products/indoor-navigation-wifi-ibeacons/

[28] "Insiteo," accessed: 2017-05-11. [Online]. Available: http://web.archive.org/web/20170511084949/http://www.insiteo.com/joomla/index.php/en/

[29] "Insiteo: Accuracy," accessed: 2017-05-11. [Online]. Available: http://web.archive.org/web/20170511084311/http://www.insiteo.com/joomla/index.php/en/our-services

[30] J. Haverinen and A. Kemppainen, "Global indoor self-localization based on the ambient magnetic field," *Robotics and Autonomous Systems*, vol. 57, no. 10, pp. 1028–1035, 2009.

[31] "IndoorAtlas," accessed: 2017-04-11. [Online]. Available: http://www.indooratlas.com/

[32] "IndoorAtlas: FAQ," accessed: 2017-04-11. [Online]. Available: http://www.indooratlas.com/faq/

[33] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, 2006, pp. 27–35.

[34] "jQuery," accessed: 2017-05-08. [Online]. Available: https://jquery.com/

[35] "W3C: SVG," accessed: 2017-04-06. [Online]. Available: https://web.archive.org/web/20170406112906/https://www.w3.org/Graphics/SVG/

[36] "Snap.svg," accessed: 2017-04-10. [Online]. Available: https://web.archive.org/web/20170410122010/http://snapsvg.io/about/

[37] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern Web Architecture," *ACM Trans. Internet Technol.*, vol. 2, no. 2, pp. 115–150, May 2002. [Online]. Available: http://doi.acm.org/10.1145/514183.514185

[38] "Jersey," accessed: 2017-04-06. [Online]. Available: https://jersey.java.net/

[39] N. Balani and R. Hathi, *Apache CXF Web Service Development: Develop and deploy SOAP and RESTful Web Services*, 1st ed. Birmingham: Packt Publishing, Limited, 2009.

[40] S. S. Sriparasa, *JavaScript and JSON essentials*, 1st ed. Birmingham: Packt Publishing, 2013.

[41] O. Oguejiofor, V. Okorogu, A. Adewale, and B. Osuesu, "Outdoor localization system using rssi measurement of wireless sensor network," *International Journal of Innovative Technology and Exploring Engineering*, vol. 2, no. 2, pp. 1–6, 2013.

[42] R. G. Akl, D. Tummala, and X. Li, "Indoor propagation modeling at 2.4 GHz for IEEE 802.11 networks," 2006.

[43] A. Goldsmith, "Stanford EE 359: Wireless Communications, Lecture 3, Shadowing, Combined Path Loss/Shadowing, Model Parameters from Data." p. 2, fall 2016, lecture given by Andrea Goldsmith at Stanford [Accessed: 2017-05-10]. [Online]. Available: http://web.archive.org/web/20170510080701/https://web.stanford.edu/class/ee359/pdfs/lecture3.pdf

[44] W. Hereman and W. S. Murphy, Jr, "Determination of a Position in Three Dimensions Using Trilateration and Approximate Distances," Colorado School of Mines, Department of Mathematical and Computer Sciences, Golden, Colorado, Tech. Rep., July 1995.

[45] T. Qiu, Y. Zhou, F. Xia, N. Jin, and L. Feng, "A localization strategy based on n-times trilateral centroid with weight," *International Journal of Communication Systems*, vol. 25, no. 9, pp. 1160–1177, 2012.

[46] B. F. D. Hähnel and D. Fox, "Gaussian processes for signal strength-based location estimation," in *Proceeding of Robotics: Science and Systems*. Citeseer, 2006.

[47] S. Pichler, accessed: 2017-05-13. [Online]. Available: http://web.archive.org/web/20170513171339/https://indoo.rs/the-slam-engine-is-here/